Naval Research Laboratory
Washington, DC 20375-5320

NRL/MR/5511--93-7422

Vcalc: a 3-Space Vector Calculator

HAROLD M. GREENWALD
FRANK J. PIPITONE

Naval Center for Applied Research in Artificial Intelligence
Information Technology Division

December 15, 1993

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave Blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 15, 1993	

4. TITLE AND SUBTITLE

Vcalc: A 3-space Vector Calculator

5. FUNDING NUMBERS

PE -62234N
TA -RS34-C74-000

6. AUTHOR(S)

Harold M. Greenwald and Frank Pipitone

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
Washington, DC 20375-5320

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL/MR/5511--93-7422

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5660

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

Vcalc is a 3-space vector calculator providing graphic displays for vector and scalar operations. Designed to operate on a Sun Workstation, the program may be run stand-alone or as a debugging tool linked to a C program. It is a straightforward and useful tool whose development was motivated by the difficulty of debugging geometric programs.

14. SUBJECT TERMS

15. NUMBER OF PAGES

16

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std 239-18
298-102

Vcalc: a 3-Space Vector Calculator

INTRODUCTION

Vcalc is a 3-space vector calculator providing graphic displays for vector and scalar operations. Designed to operate on a Sun Workstation, the program may be run stand-alone or as a debugging tool linked to a C program. It is a straightforward and useful tool whose development was motivated by the difficulty of debugging geometric programs.

STAND-ALONE MODE

To run Vcalc stand-alone, simply type **vcalc**. The screen as shown in Figure 1 will appear.

Vcalc COMMANDS

To select an operation, type the corresponding acronym. For example, to receive instructions for computing the scalar triple product of three vectors, type **tp**. Following is a brief description of each command:

as Store the sum of two scalars into the output scalar slot.

av Store the sum of two vectors into the output vector slot.

cl Clear all vector and scalar slots.

cp Store the cross product of two vectors into the output vector slot.

cs Copy a new scalar into a scalar slot, or copy an existing scalar from one slot into another.

cv Copy a new vector into a vector slot, or copy an existing vector from one slot into another.

d Store the distance between two vectors into the output scalar slot.

dp Store the dot product of two vectors into the output scalar slot.

ds Store the quotient of two scalars into the output scalar slot.

dsqr Store the distance squared between two vectors into the output scalar slot.

Manuscript approved July 20, 1993.

ms	Store the product of two scalars into the output scalar slot.
mvs	Store the product of a vector and scalar into the output vector slot.
nv	Store a normalized vector into a specified output vector slot.
q	Exit Vcalc
rv	Rotate a vector N degrees around another vector and store the result into a specified output slot.
rx	Rotate a vector N degrees around the X axis and store the result into a specified output slot.
ry	Rotate a vector N degrees around the Y axis and store the result into a specified output slot.
rz	Rotate a vector N degrees around the Z axis and store the result into a specified output slot.
sc	Adjust the graduation spacing (i.e the scale of the graph).
ss	Store the difference between two scalars into the output scalar slot.
sv	Store the difference between two vectors into the output vector slot.
tp	Store the scalar triple product of three vectors into the output scalar slot.
vm	Compute the length of a vector and store the result into the output scalar slot.
vsq	Store the dot product of a vector with itself into the output scalar slot.
x	View the vector display along the X axis.
y	View the vector display along the Y axis.
z	View the vector display along the Z axis.

Whereas several vector operations allow the user to specify in which "Vector List" slot the result is to be placed, the output vector will always appear in slot 0 (marked "Output Vector") by default. The same concept applies to scalars.

DEBUGGING C PROGRAMS

To run Vcalc as a debugging tool, two source files need to be included within the compilation stream, vcalc.c and vc_lib.c. Pixrects and Math are the only libraries required. An example compilation might look like:

cc myprog.c vcalc.c vc_lib.c -o myprog -lm -lpixrect

Vcalc involves the use of the two macros ev(vector), es(scalar), and the function vc():

```
#include <vcalc.h>
ev(vector)                      /* ev is used to load a vector into Vcalc.*/
double *vector;                 /* The input argument points to an array of three doubles. */

es(scalar)                      /* es is used to load a scalar into Vcalc.*/
double scalar;                  /* The input argument contains a scalar. */

vc();                           /* vc() activates Vcalc along with all vectors and scalars
                                   previously loaded  with ev() and es(). */
```

The following program provides a simple example of how Vcalc is called from a C program. Note the use of the above macros in porting program data into Vcalc.

```
#include <stdio.h>
#include "vcalc.h"
double vec1[3], vec2[3], vec3[3],vec_array[11][3];
double scal1, scal2[12];
main()
{
        vec1[0]=3.3; vec1[1]=4.4; vec1[2]=5.5;
        ev(vec1);               /*Add vec1 to the (currently empty) vector list */
        vec2[0]=23.273; vec2[1]=-10.4; vec2[2]=15;
        ev(vec2);               /* Add vec2 to the vector list */
        vec_array[5][0]=40;
        vec_array[5][1]=9.89;
        vec_array[5][2]=-20.41;
        ev(vec_array[5]);       /* Add vec_array[5] to the vector list */
        vec3[0]=-28.00; vec3[1]=-.38; vec3[2]=4.00;
        ev(vec3);               /* Add vec3 to the vector list */
        scal1=123.987;
        scal2[6]=3;
        es(scal1);              /* Add scal1 to the (currently empty) scalar list */
        es(scal2[6]);           /* Add scal2 to the scalar list */
        vc();                   /* Bring up Vcalc */
}
```

This example results in the screen image shown in Figure 2. The figure also illustrates the selection of the rz command. Recall that the rz command rotates a vector around the Z axis.

The four vectors and the two scalars passed to vc() via ev() and es() are listed in the "Vector List" and "Scalar List." respectively. The vectors are drawn and labeled according to their index into the vector list. The standard vector input prompt is:

enter vector (0 thru 10 -or- 11 for a new vector):

0-10 refers to the vector list index. A new user-specified vector may be entered by selecting 11. Vector number 4 has been selected for this rotation. In our test program, the variable name vec3 is displayed in vector slot 4 along with its X,Y,Z values. Vector slot 6 has been selected as the output location and a 90 degree rotation is specified. The result of rotating (-28,-38,4) 90 degrees around the Z axis is shown in Figure 3.

Notice in Figure 3 that the output from the previous rotation operation was stored in vector slot 6 without a name. Only those vectors and scalars passed directly from the calling program have their names listed (vec1, scal2 [6], etc.). Output values from Vcalc operations as well as vectors and scalars entered in by hand are listed without variable names.

Figure 3 shows that we have selected the *rotate vector* command (**rv**) for our next example. Vector 2 will be rotated around vector 1 by 180 degrees and the output will be stored in vector 2. Variable vec2 will hold the result of rotating the previous contents of vec2 180 degrees around vec1.

The current graduation spacing, as shown directly beneath the graph, indicates that each line on the graph represents one unit of measure. This value is adjustable. For example, we can set each line equal to five units by simply selecting *adjust scaling* (**as**) and entering 5 at the prompt as shown in Figure 4.The resulting screen is shown in Figure 5. Figure 5 also shows us instructing Vcalc to normalize vector 4 (i.e. vec3) and place the result in vector slot 8.

In Figure 6 we take the cross product of vector 8 (i.e., the output of the previous operation) and vector 1. The result is stored in the default Output Vector slot (slot 0). Besides the cross product, Figure 7 shows that we have set the graduation spacing back to 1. We can view the current vectors along the Y axis by entering Y (Fig. 8).

Several scalar operations are available. Figures 9 and 10 show us taking the dot product of vectors 3 and 1, with the output going into the default output scalar slot (scalar slot 0).

As with vectors, we always have the option of using new scalar data by entering 11 at the following prompt:

enter scalar (0 thru 10 -or- 11 for a new scalar):

We will expand the capabilities of Vcalc as the need arises. Since one of our major interests is the interpretation of range images, one possible direction for expansion is the fitting of simple surfaces to range data, and the computation of angular and linear displacements between such structures as planes, lines, dihedral edges, trihedral vertices, etc.

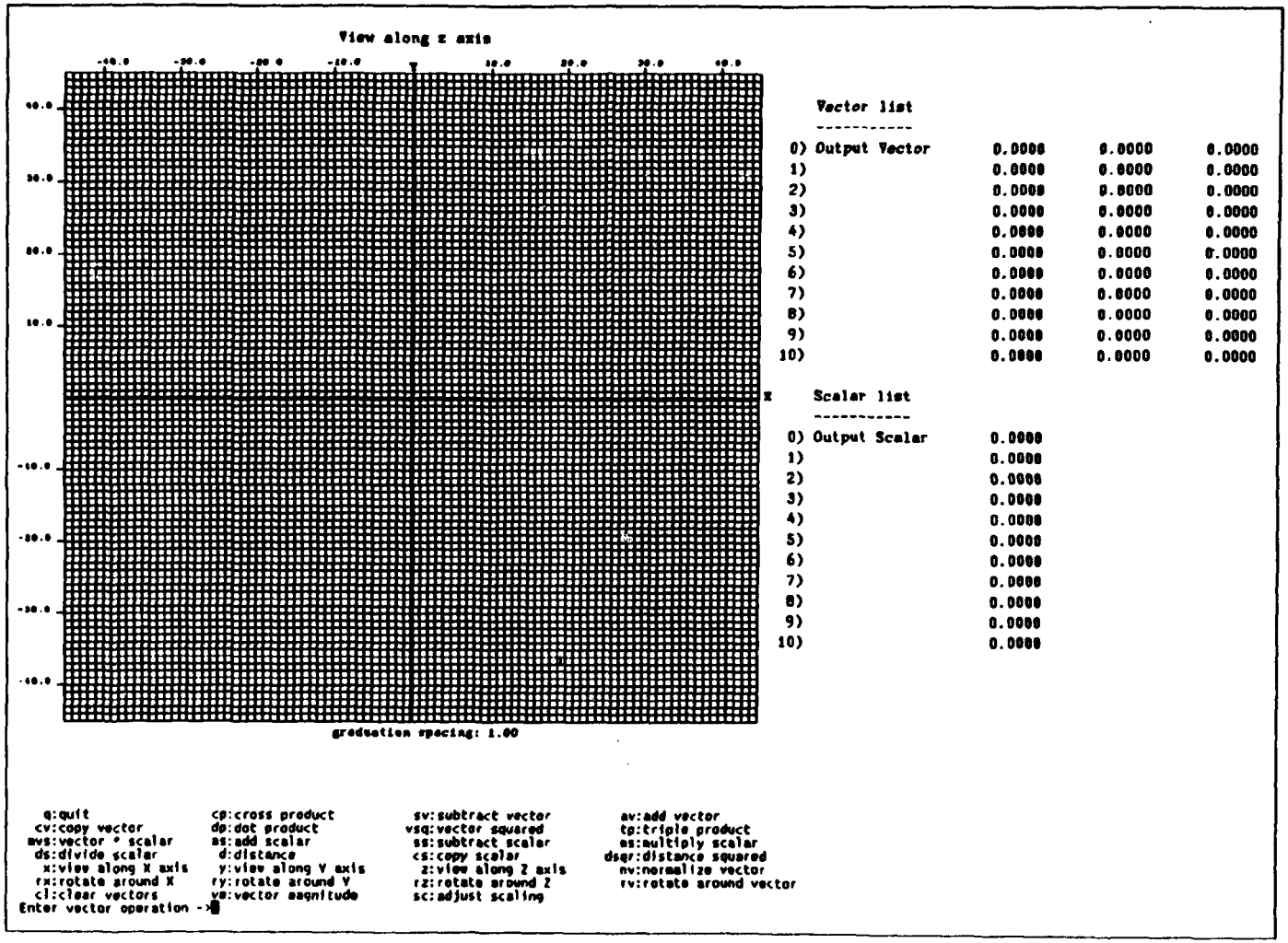

Figure 1. Vcalc's opening screen - In this example, no vector or scalar data has been passed to Vcalc.

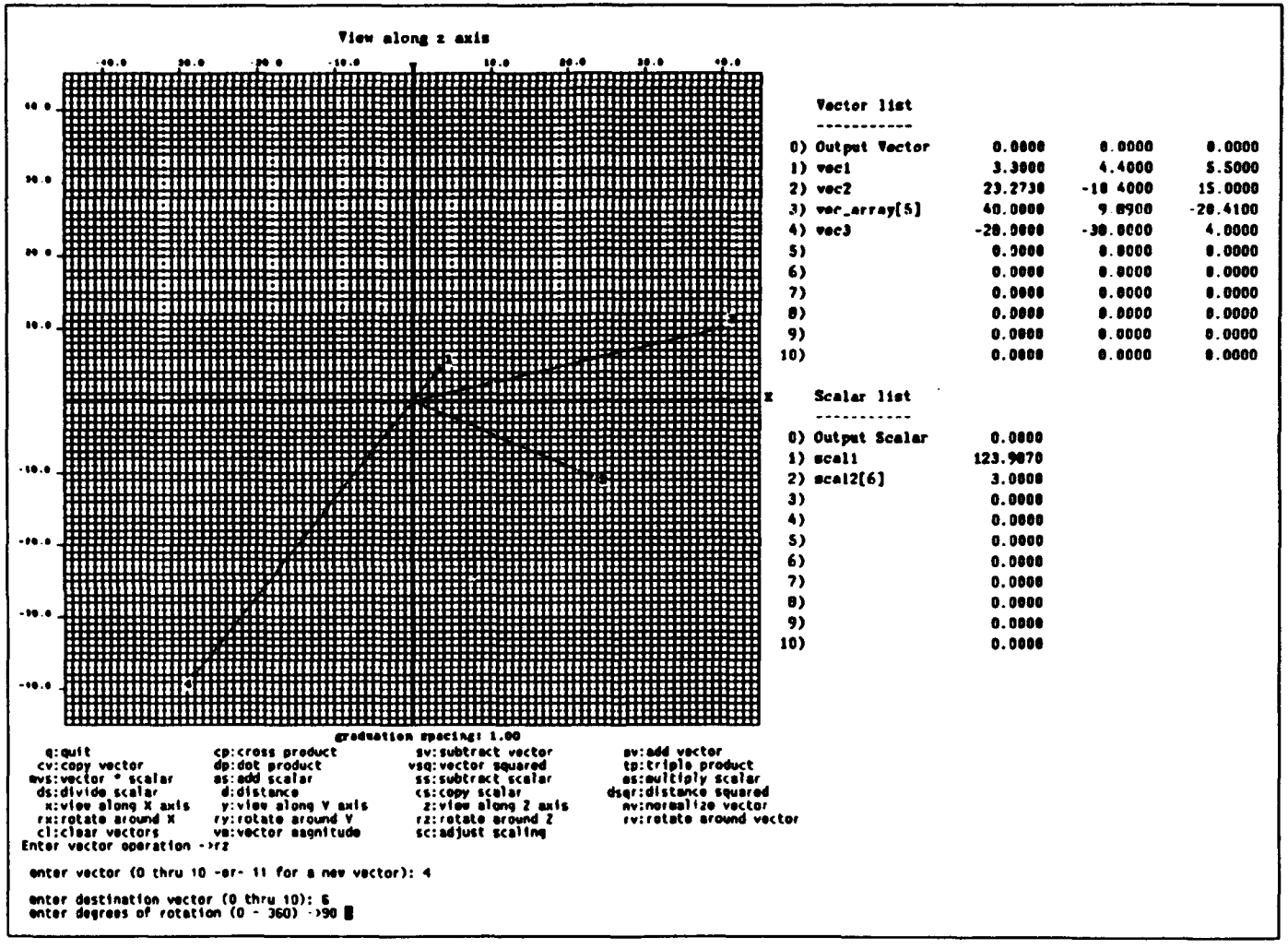

Figure 2. The user is instructing Vcalc to rotate vec3 90 degrees around the Z axis, and place the result in vector slot number 6.

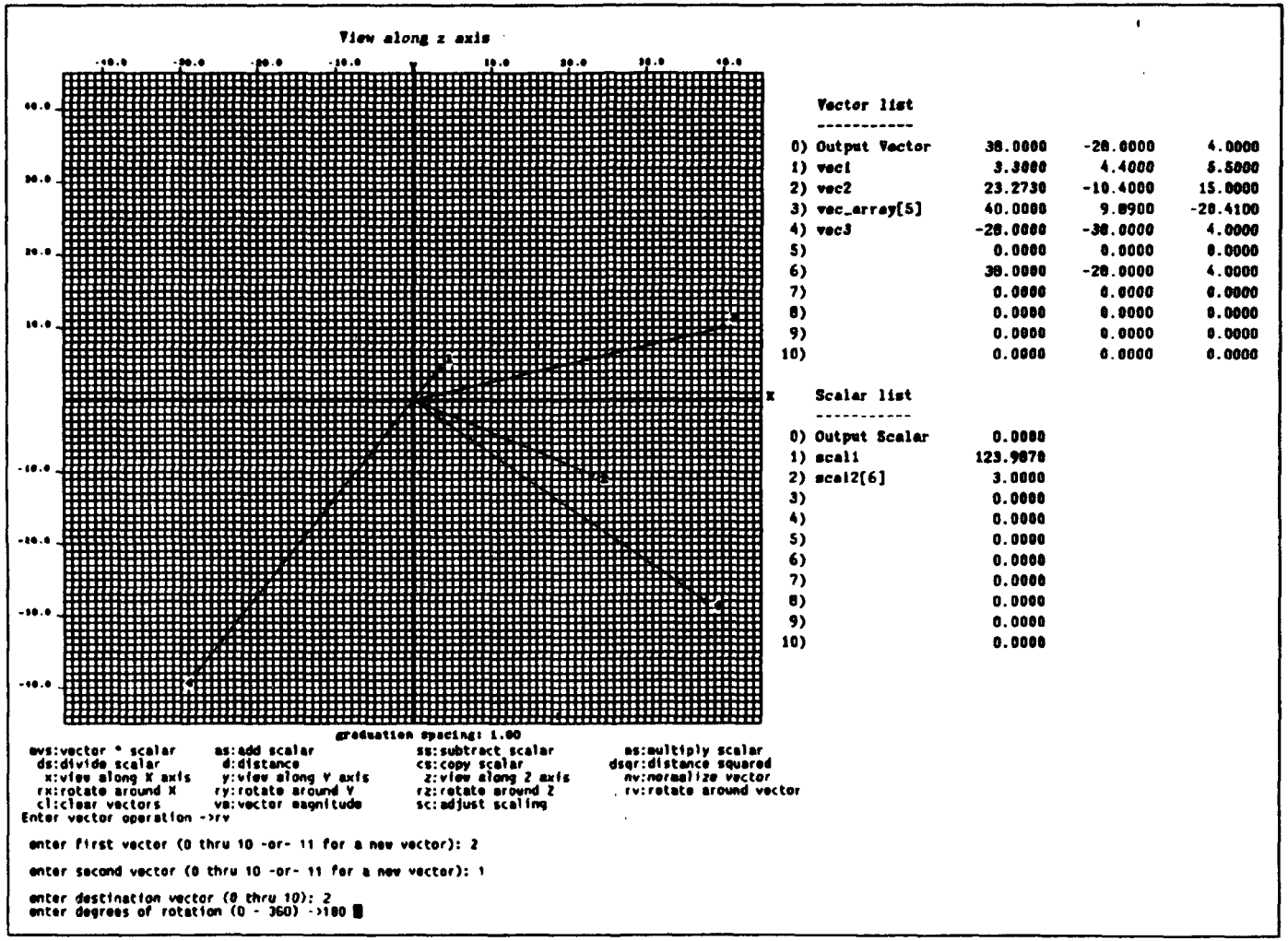

Figure 3. The user is instructing Vcalc to rotate vec2 180 degrees around vec1, and place the result in vector slot number 2.

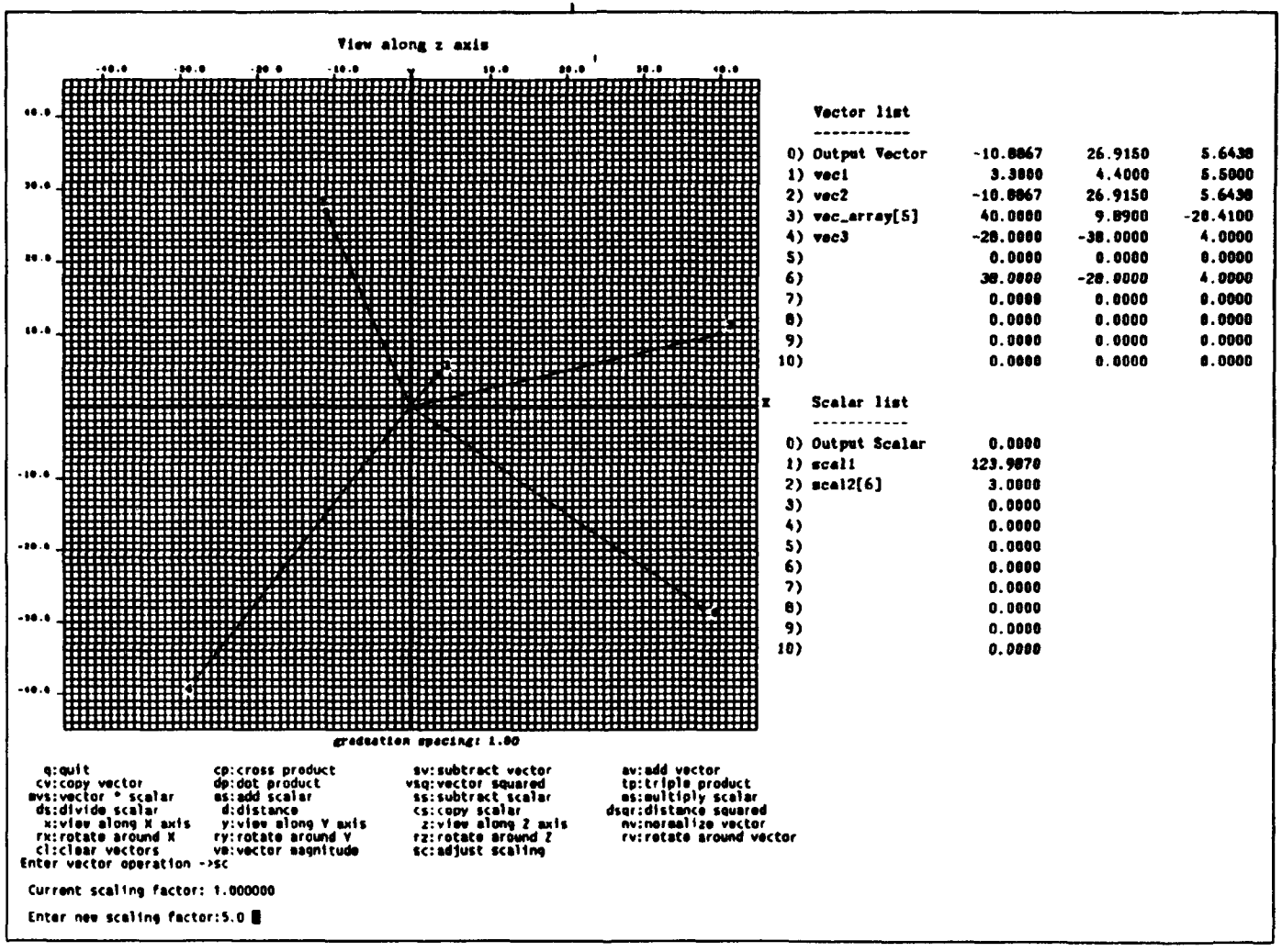

Figure 4. Vector number 2 (vec2) has been rotated 180 degrees around vec1. The graduation spacing (scaling) is being adjusted from 1.0 to 5.0.

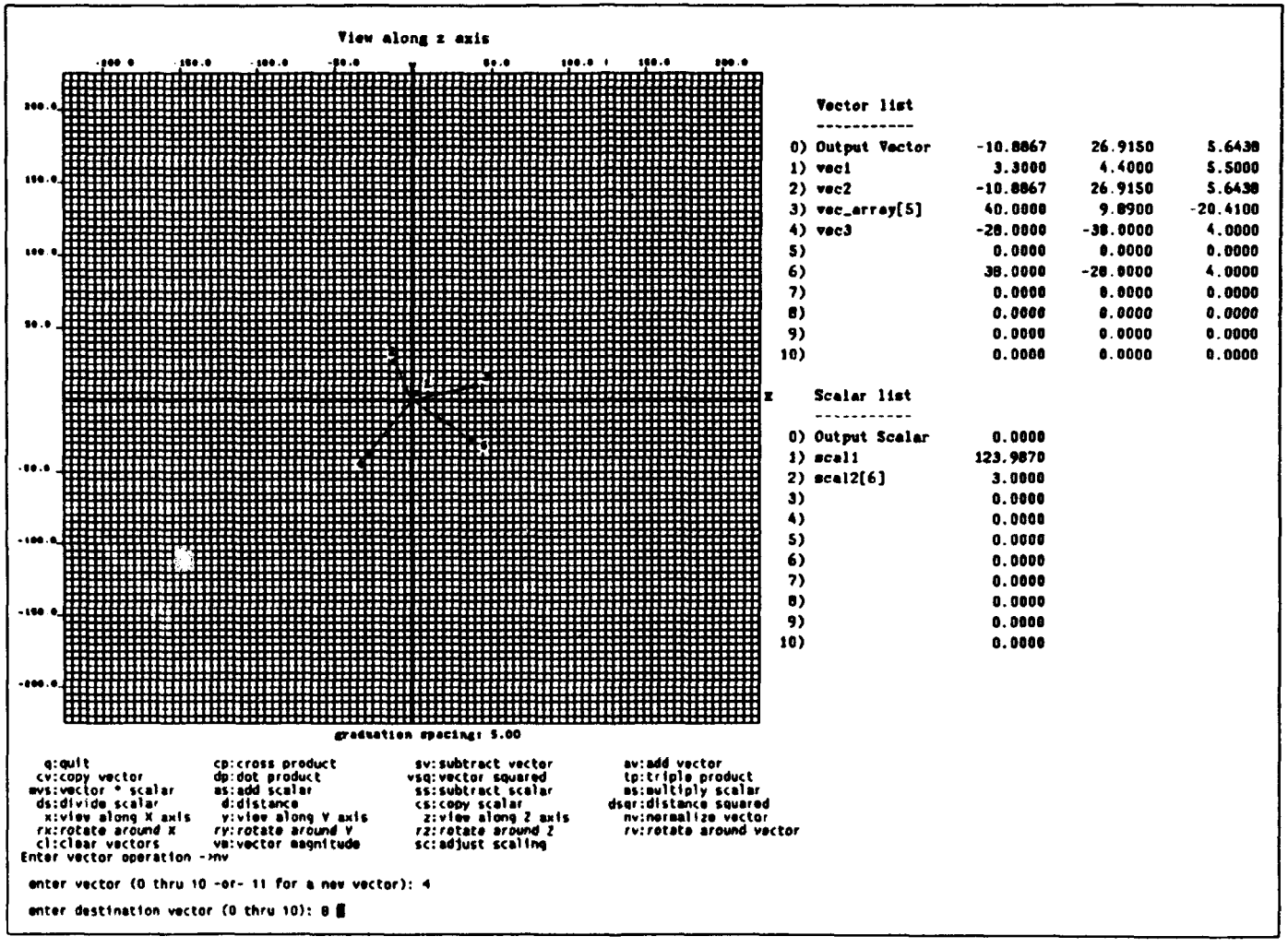

Figure 5. With graduation spacing now equal to 5.0, Vcalc is being instructed to normalize vector number 4 (vec3). The output will be placed in vector slot 8.

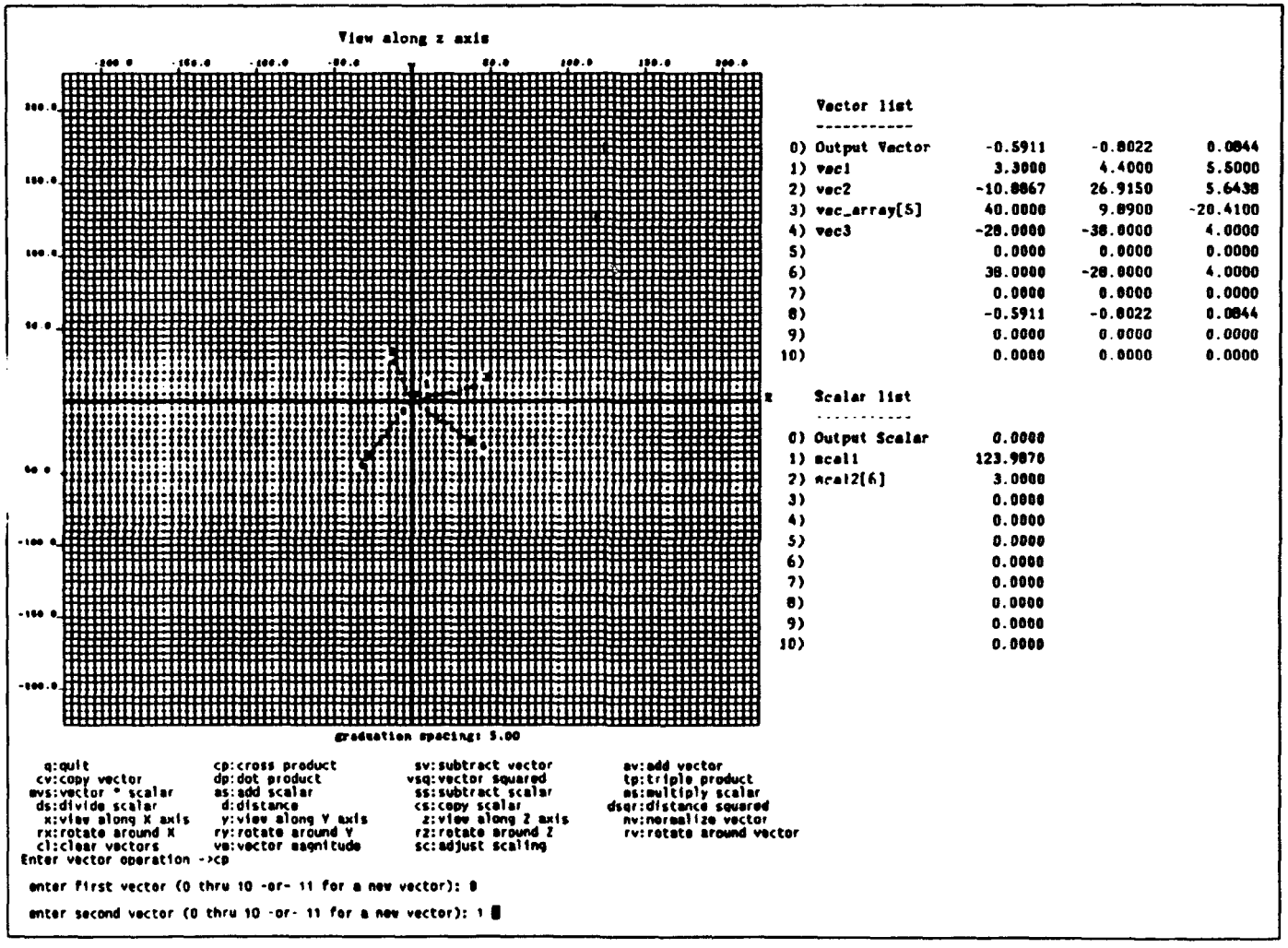

Figure 6. The user is instructing Vcalc to take the cross product of vector number 8 and vector number one (vec1). The output will be automatically stored in vector slot 0.

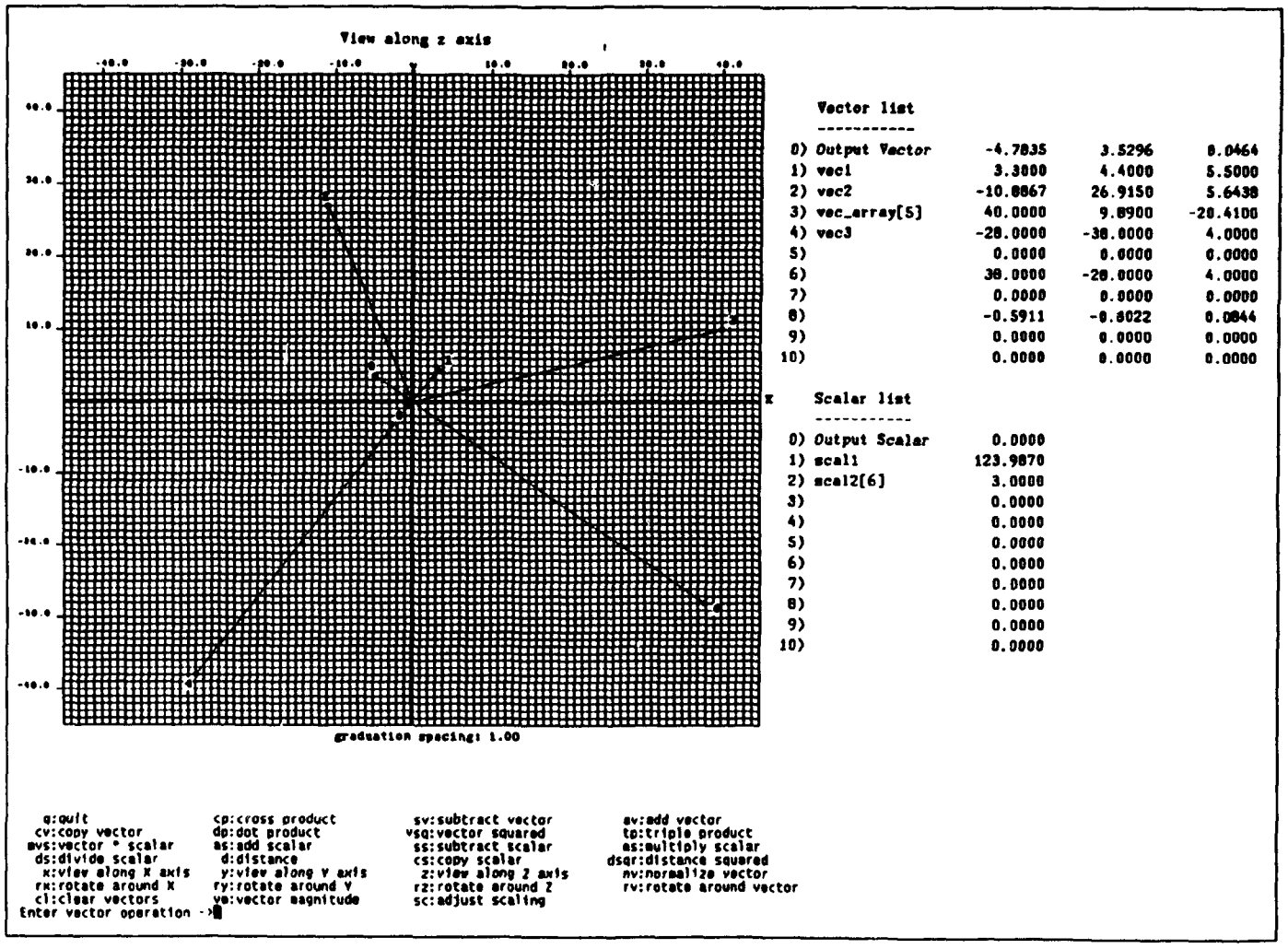

Figure 7. Graduation spacing has been set back to 1.0, and the cross product of vector number 8 and vector number one (vec1) is now stored in vector slot 0 as well as being graphically displayed.

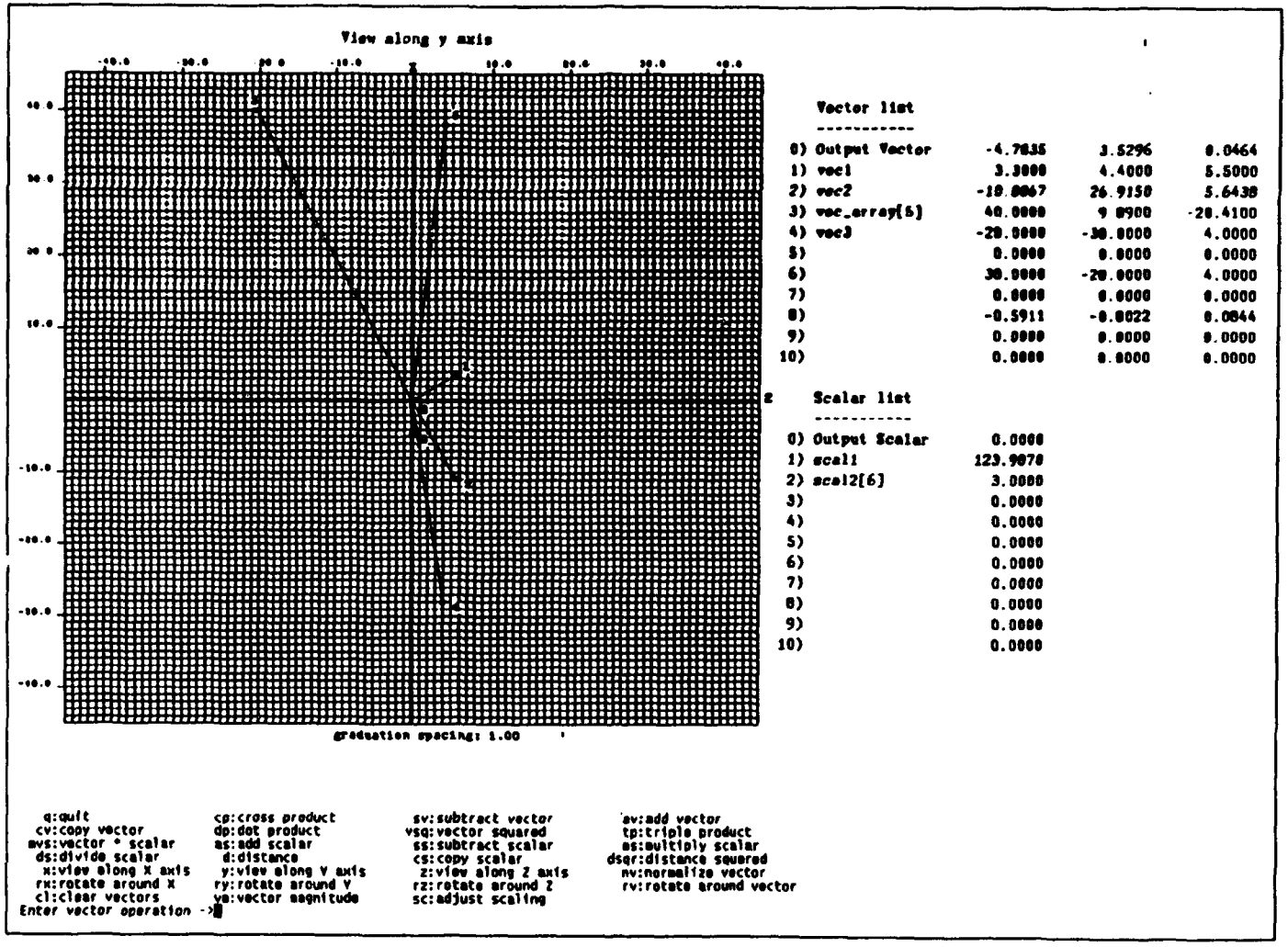

Figure 8. By selecting Y, we can view the current vector list along the Y axis.

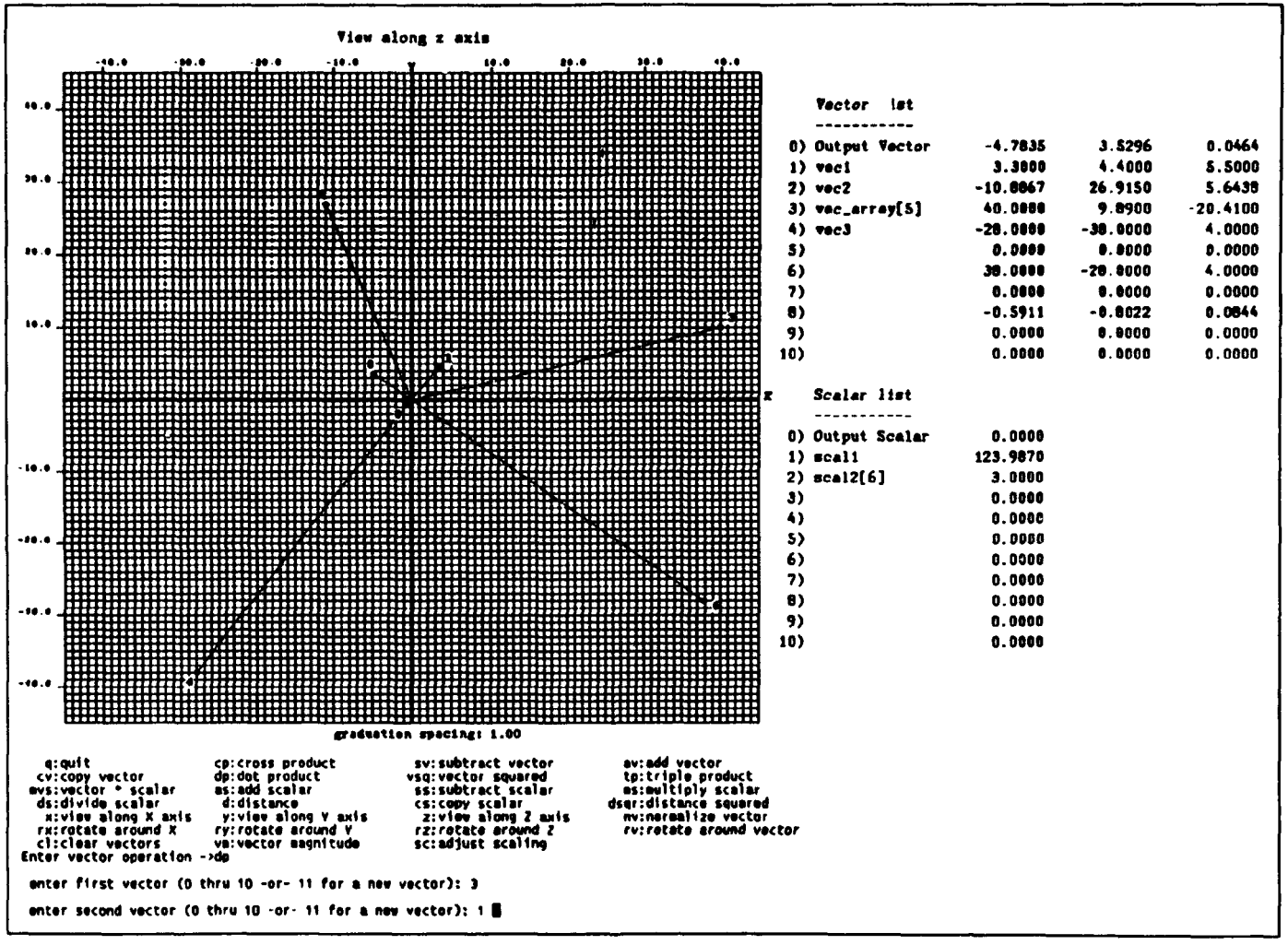

Figure 9. By having selected Z (not shown), we are once again viewing the vector list along the Z axis. The user is now instructing Vcalc to compute the dot product of vector number 3 (vec_array[5]) and vector number 1 (vec1). The result will be stored in scalar slot 0.

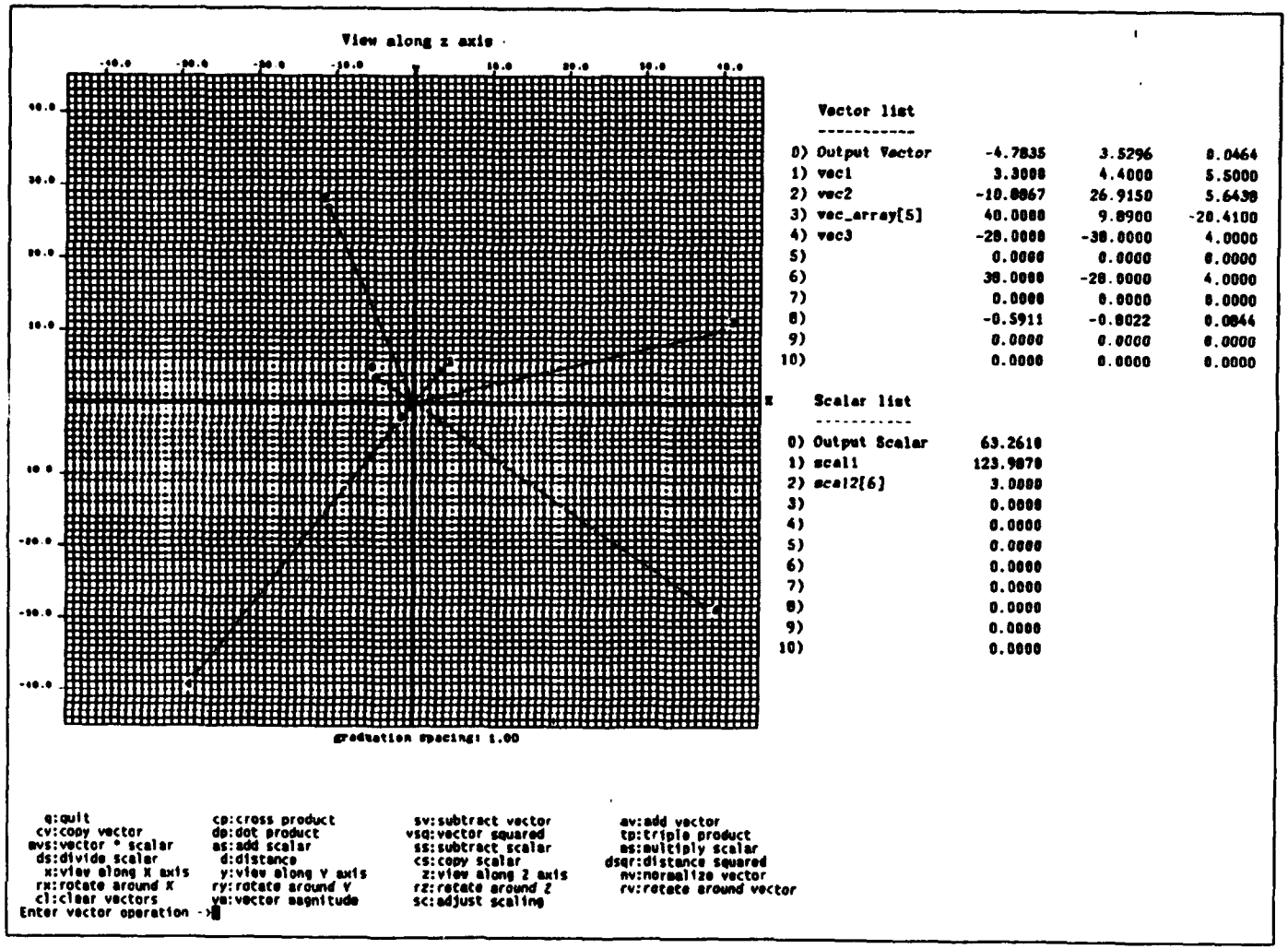

Figure 10. The dot product of vec_array[5] and vec1 is now stored in scalar slot 0.